Lean cohomology computation for electromagnetic modeling

Paweł Dłotko[1], Bernard Kapidani[2], Ruben Specogna[2]

[1]Department of Mathematics, Swansea University, Singleton Park, Swansea SA2 8PP, UK
[2]Polytechnic Department of Engineering and Architecture (DPIA), Università di Udine, 33100 Udine, Italy

Solving eddy current problems formulated by using a magnetic scalar potential in the insulator requires a topological pre-processing to find the so called first cohomology basis of the insulating region, which may result being very time consuming for challenging industrially driven problems. The physics-inspired Dłotko–Specogna (DS) algorithm was shown to be superior to alternatives in performing such a topological pre-processing. Yet, the DS algorithm is particularly fast when it produces as output not a regular cohomology basis but a so called lazy one, which contains the regular one but it keeps also some additional redundant elements. Having a regular basis may be advantageous over the lazy basis if a technique to produce it would take about the same time as the computation of a lazy basis. In literature such a technique is missing.

This paper covers this gap by introducing modifications to the DS algorithm to compute a regular basis of the first cohomology group in practically the same time as the generation of a lazy cohomology basis. The speedup of this modified DS algorithm with respect to the best alternative reaches more than two orders of magnitudes on challenging benchmark problems. This demonstrates the potential impact of the proposed contribution in the low-frequency computational electromagnetics community and beyond.

***Index Terms*—eddy currents, magnetic scalar potential, cuts, cohomology, first de Rham cohomology group**

I. Introduction

EDDY-current problems are typically modeled by considering a continuous domain D being a topologically trivial 3-manifold with boundary embedded in $\mathbb{R}^3$ [1]. For the purpose of computations, the domain D is meshed with a cell complex $\mathcal{K}$ which decomposes into two sub-complexes: $\mathcal{K}_a$ and $\mathcal{K}_c$ representing the insulating and the conducting sub-regions of D, respectively. For eddy current formulations based on the magnetic scalar potential in $\mathcal{K}_a$, a set of representatives that span the first cohomology basis [1] of $\mathcal{K}_a$, denoted as $H^1(\mathcal{K}_a)$, are required, see for example [2]-[4]. The recent attempts to efficiently solve eddy-current problems based on this formulation have allowed the computational topology [1] in general, and computational cohomology in particular, to made its way into commercial [5], [6] and research [7], [8] electromagnetic simulation software.

In principle, the algorithms to compute homology and cohomology have been established along with the theories themselves, but they are slow. Thanks to illuminated engineers like Kotiuga [2], [9]-[11] new, more efficient algorithms started to be emerge. Yet, those algorithms, although bringing considerable progress, could not yet be considered practical for industrial engineering. We also mention the algorithm described in [3] which in some cases may output a collection of edges that contain the union of the supports of cohomology generators. The author of [3] never proved or claimed that the algorithm produces a cohomology basis as output. Moreover, how the algorithm is documented inside [3] does not enable to solve the issues raised in [6].

Manuscript received June ??, 2017; accepted ??. Date of current version June 11, 2017. Corresponding author: R. Specogna (e-mail: ruben.specogna@uniud.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMAG.2013.2281076

During the last years our effort was focalized to cut down the time required by the topological pre-processing by combining and improving various techniques introduced in literature by Webb and Forghani [18], Kotiuga [11] and Hiptmair [14]. Our main contribution to the field is the Dłotko–Specogna (DS) algorithm [12], [13] that admits certain features that distinguishes it from all algorithms proposed so far including [11] and [8]:

1) The computation uses the complex $\mathcal{K}$ and not just one of the sub-complexes $\mathcal{K}_a$ or $\mathcal{K}_c$.
2) The computations are initially performed at the interface $\mathcal{K}_a \cap \mathcal{K}_s$ with a combinatorial algorithm, in place of a standard algebraic algorithm applied on the whole $\mathcal{K}_a$ like [11] and [8]. Note that the cardinality of $\mathcal{K}_a \cap \mathcal{K}_s$ is typically considerably smaller than the one of $\mathcal{K}_a$.
3) The cuts are then propagated from $\mathcal{K}_a \cap \mathcal{K}_s$ to $\mathcal{K}$ with a general version of Webb–Forghani algorithm [18].

The DS algorithm has been proved to be superior to all the competing approaches to perform the required topological pre-processing in terms of speed and memory consumption (see [7]), while keeping the full generality. The output provided by the fastest version of the DS algorithm consists of lazy generators which span the $H^1(\mathcal{K}_a)$ space, but are not linearly independent. After plugging the lazy cohomology basis inside electromagnetic simulation software, the resulting system of equations is not full of rank, yet it gives the correct solution of the problem in terms of, for example, induced current density. When requesting a regular $H^1(\mathcal{K}_a)$ basis one can use a slower and more complicated version of the DS algorithm, described in [12] and recalled in Section II, to compute it.

This paper introduces an important extension in the DS algorithm [12], [13] to efficiently compute a regular basis of $H^1(\mathcal{K}_a)$ in about the same time the original DS algorithm produced a lazy cohomology basis.

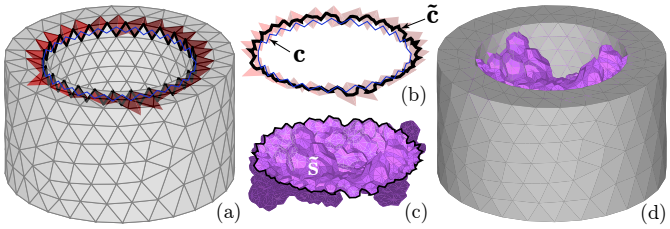

Fig. 1. (a) A solid 2-torus conductor. The thick edges represent the support of a representative of the cohomology generator of the conductor's boundary. The dark triangles represent the thinned current. (b) The dual edges, which are dual to thinned current faces form a cycle $\tilde{c}$ in the dual complex. (c) The dual cycle $\tilde{c}$ is the boundary of a (possibly self intersecting) oriented surface $\tilde{s}$ on the dual complex. (d) $\tilde{s}$, restricted to $\mathcal{K}_a$, is the dual of a cohomology generator of $\mathcal{K}_a$.

The paper is organized as follows. Section II surveys the original DS algorithm. Section III presents the modified DS algorithm called later *lean DS algorithm*. Section IV presents experimental results to validate the lean DS algorithm and to compare it in terms of speed and memory consumption to other state-of-the-art implementations. Finally, in Section V, the conclusions are drawn.

II. DS ALGORITHM

In this section we recall the DS algorithm firstly introduced in [12]:

1) The first cohomology group generators of the discrete surface $\mathcal{S} = \mathcal{K}_c \cap \mathcal{K}_a$ (see Fig. 2a) are computed with a linear time worst-case complexity algorithm. Note that $\mathcal{S}$ may have more than one connected component.
2) *Thinned currents* are found by pre-multiplying the representatives of generators obtained in step 1 by the incidence matrix $\mathbf{C}_c$ between face and edge pairs [12]. The support of a thinned current of a toric conductor is represented in Fig. 2a by the dark faces.
3) Finally, a vectorialized version of the Extended Spanning Tree Technique (ESTT) algorithm [15] is run on the whole complex $\mathcal{K}$ for all thinned currents obtained in step 2 at the same time. The ESTT algorithm is a general version of the Webb–Forghani (WF) iterative algorithm [18] used to obtain a discrete field whose discrete curl is assigned (in our case to the curl of the thinned current). The output of the ESTT restricted to $\mathcal{K}_a$ form the cohomology generators (Fig. 2d) [12].

From now on we also use the concept of dual complex and exploit the dualities between the original and this dual complex [16]. The dual of a thinned current forms a 1-cycle $\tilde{c}$ on the dual complex, see the thick edges in Fig. 2b. An important interpretation that is going to be used later is that the ESTT is computing an oriented discrete surface $\tilde{s}$ on the dual complex (possibly self-intersecting) having $\tilde{c}$ as boundary, see Fig. 2c. A more formal demonstration of this fact can be found in [17].

The procedure recalled above describe the standard DS algorithm that computes a lazy cohomology basis. It can be modified to compute a regular cohomology basis by adding the following step just after step 2 of the DS algorithm.

2.5 The Hiptmair–Ostrowski (*HO*) technique [14] adapted as described in [12] is used to construct a new set of representatives of cohomology generators of $\mathcal{S}$ from the ones obtained in step 1. Half of the new generators are trivial in $\mathcal{K}_c$ and the remaining half are trivial in $\mathcal{K}_a$. A cocycle is trivial if a cycle dual to it bounds a surface in the considered sub-complex. In the further steps we use only the generators which are trivial in $\mathcal{K}_a$.

We focus now on the additional step 2.5 of the algorithm. The *HO* technique finds the required change of the $H^1(\mathcal{S})$ basis by computing the null-space basis of a (typically sparse) matrix of dimension $2g \times 2g$, g being the genus of $\mathcal{S}$. The matrix stores all the mutual linking numbers between the cycles dual in $\mathcal{S}$ to the representatives of $H^1(\mathcal{S})$ generators obtained in the step 1 (see c in Fig. 2b) and the dual of the thinned currents from the step 2 (see $\tilde{c}$ in Fig. 2b) [12]. For complicated examples, the bottleneck of the whole process is the computation of this matrix. To put it into perspective: in the last benchmark problem proposed in this paper—which arises from a practical engineering problem—g is 1621, which means that more than 10 millions of linking numbers have to be computed to construct the matrix. This yields to an insurmountable bottleneck due to the *HO* technique even when taking advantage of parallelism in the computation of linking numbers. To avoid this problem in the original DS algorithm we do not use the step 2.5 [12], [13]. Yet, a standard cohomology basis is computationally attractive given that it enables an easy enforcement of current sources (more on this in Section IV-B), it reduces the number of unknowns in the linear system and also produces a full-rank system matrix which guarantees the uniqueness of the system solution in terms of potentials also. To obtain a standard cohomology basis we introduce in the next Section a new technique to remove this computational obstruction in a negligible computational cost.

III. LEAN DS ALGORITHM

The new idea presented in this contribution eliminates the bottleneck in the computation of the matrix of linking numbers by taking advantage of a simple observation: lazy cohomology generators can be used to compute the matrix of linking numbers in linear time w.r.t. the sum of the cardinality of the support of the generator's representatives. To be more precise: when constructing the matrix of linking numbers, our aim is to compute the linking number between $\tilde{c}$, a cycle dual to a thinned current, and another cycle d. This is equivalent to the computation of a dot product between any dual (possibly self-intersecting) oriented surface bounded by $\tilde{c}$ and the cycle d, see [21]. This equivalent method was used in a different context for example in [19]. The idea is therefore to use the representatives of the lazy cohomology generators as the surfaces to conduct those computations.

In the presented case, the cycles d for which the linking number is computed are cycles in $\mathcal{S}$ which are $H_1(\mathcal{S})$ generators dual to the $H^1(\mathcal{S})$ basis obtained in the DS algorithm in such a way that c and d are in the same homology class by construction, see Fig. 2ab. Note that those homology generators are obtained from the combinatorial algorithm to

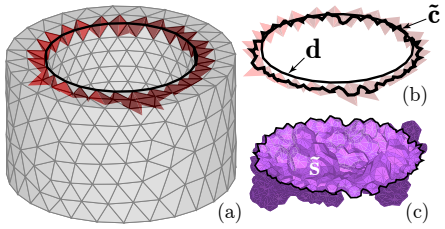

Fig. 2. (a) A solid 2-torus conductor. The thick edges represent the support of a representative of the homology generator of the conductor's boundary. The dark triangles represent the thinned current. (b) Cycle c of Fig. and cycle d are in the same homology class. (c) The dual cycle $\tilde{c}$ is the boundary of a surface $\tilde{s}$ on the dual complex.

compute the generators of $H^1(\mathcal{S})$ used at the step 1 of DS algorithm at no additional cost [12], [20].

The *lean DS* algorithm then operates as follows:

1) The first homology group generators of the discrete surface $\mathcal{S} = \mathcal{K}_c \cap \mathcal{K}_a$ (see Fig. 2a) are computed with a linear complexity algorithm.
2) *Thinned currents* are computed starting from all homology generators of $\mathcal{S}$ by using a technique similar explained in the Section II.
3) Vectorialized version of the ESTT algorithm is run for all thinned currents.
4) For every cycle d being the representative of a $H_1(\mathcal{S})$ generator, compute in a vectorialized way, its intersection with all supports of the representatives of the lazy cohomology generators. The complexity of this operation for a cyce d is gn, where g is the number of lazy cohomology generators, and n is the number of edges in d. This way, fill in the matrix of linking numbers and compute the null space basis of it.
5) The adapted *HO* technique is applied. The change of basis found in the previous step is applied to the lazy cohomology generators after restricting them to $\mathcal{K}_a$.

IV. Numerical results

The algorithm presented in this paper is implemented as a part of the TOPOPROCESSOR C++ package [7]. To show how the lean DS algorithm enhancement in TOPOPROCESSOR further advocates for the use of the package, we perform the topological preprocessing required by the h-oriented eddy current formulations in the three industrial test cases already used in [7]. The reader is invited to consult [7] for the pictures of the benrchmarks given that there is no space to represent them here. The performance of the lean DS algorithm is compared with the one of an output equivalent tool provided in GMSH [8], which is an efficient implementation of the standard paradigm based on reducing the complex and computing the Smith Normal Form (SNF) of the reduced matrix.

In what follows, t_H denotes the total wall time (in seconds) needed by the two competing tools to compute a regular cohomology basis. Furthermore, for every test, the wall time needed by the standard DS algorithm implemented in TOPOPROCESSOR for to the computation of the *lazy* basis is added between brackets. All computations are performed on a workstation with a 12Core-Xeon E5-2687Wv4 processor equipped with 192 GB of RAM. We also indicate with which mesh generator (GMSH or NETGEN [22]) the original tetrahedral mesh has been generated for each test.

A. b1: heat exchanger

TABLE I
HEAT EXCHANGER BENCHMARK

	Lean DS	GMSH
N. of tets in $\mathcal{K}$	10 445 468	10 445 468
N. of tets in $\mathcal{K}_a$	8 820 579	8 820 579
Meshing time (GMSH) [s]	272.6	272.6
t_H [s]	53.9 (53.2)	680.2
Peak RAM usage [GB]	9	27.7

Quality controls and maintenance of shell and tube heat exchangers is routinely performed with eddy current non-destructive testing. The first test comprises the complement of a heat exchanger with respect to a box. The time required to perform the topological preprocessing is represented in Tab. I together with some information on the meshes.

B. b2: magnetic induction tomography (MIT)

TABLE II
MAGNETIC INDUCTION TOMOGRAPHY BENCHMARK

	Lean DS	GMSH
N. of tets in $\mathcal{K}$	7 976 328	7 976 328
N. of tets in $\mathcal{K}_a$	7 484 064	7 484 064
Meshing time (NETGEN) [s]	210	210
t_H [s]	38.2 (37.9)	6 723.6
Peak RAM usage [GB]	7.0	23.3

Magnetic induction tomography (MIT), also known as eddy current non-destructive testing, typically uses an array of coils that surrounds a conductive rod to be inspected.

The classical way to enforce the current in a torus-shaped coil is to fix (by usual boundary conditions techniques) the degree of freedom corresponding to a cohomology generator (i.e. the *independent currents* in [12]). This is possible only if a cohomology generator is in one-to-one correspondence with a toric coil. This means that, in this application, not all cohomology basis are useful and, therefore, a basis selection has to be performed. A nice feature of the DS (and lean DS) algorithm is that by design it produces a basis suitable for imposing current sources. This is due to the generators being computed from the boundary of each conductor independently.

On the contrary, there is no known mean to include natively this basis selection in algebraic methods as the ones used inside GMSH. A possible, but costly, way out is to perform computations separately for each coil, inspired from [23]. To be precise, in the j-th computation just the j-th coil is considered as conductor, whereas all others are considered as insulator. The wall time needed by GMSH to compute cohomology basis with the described basis selection has been added in Table II.

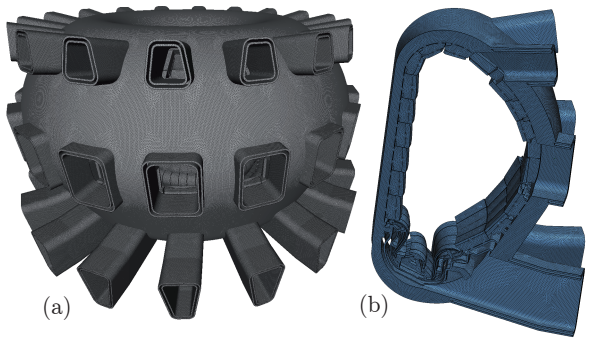

Fig. 3. (a) The geometry of the considered conductive structures of an ITER-like nuclear fusion reactor. (b) One eighteenth of the geometry.

TABLE III
NUCLEAR FUSION REACTOR BENCHMARK

	Lean DS	GMSH
N. of tets in $\mathcal{K}$	26 648 351	26 648 351
N. of tets in $\mathcal{K}_a$	13 225 740	13 225 740
Meshing time (GMSH) [s]	1116	1116
t_H [s]	221.2 (220.8)	17 839
Peak RAM usage [GB]	25.2	64

C. b3: ITER-like nuclear fusion reactor

The last test considers the complement of the conductive structures of an ITER-like nuclear fusion device (see Fig. 3a) with respect to a box which represents the insulating region. The conductor is formed by gluing together 18 structures as the one in Fig. 3b. The number of mesh elements and the topological features render the topological pre-processing for this benchmark particularly challenging. In fact, we expect to extract 1621 generators of the first cohomology group of the insulating region, which corresponds to 3242 lazy generators.

TOPOPROCESSOR code has been able to compute all generators in less than 4 minutes of total computing time, whereas GMSH terminated after approximately 5 hours on the same workstation. Details on the mesh and on timings are shown in Tab. III. It is also interesting to note that GMSH has a higher memory consumption. If it was run on a machine with 32 GB of memory (which is a very realistic scenario in the present industrial environment) it would need to use memory swapping for a problem of this complexity. Indeed, simulations performed on a virtual machine with reduced memory show an additional increase of one order of magnitude in the speedup provided by the lean DS algorithm (which is already roughly a factor of 80 for the setup of Tab. III).

V. CONCLUSION

Lean cohomology computation is a provably general method to compute a first cohomology basis in the described setting. Practical comparisons with other state-of-the-art competing software show that it is the fastest method to date in performing such computations. This well motivates in our opinion the interest in the algorithm proposed in this paper. It is unlikely that further sensible reduction of the computational burden of cohomology computations will be achieved in the future (apart from obvious technology driven computational power improvements). Finally, we remark that the topological pre-processing is a fraction of meshing time in the case of TOPOPROCESSOR, whereas it ends up being the bottleneck of the whole simulation chain in the case of GMSH.

REFERENCES

[1] K. Itō, *Encyclopedic Dictionary of Mathematics*, 2nd ed., MIT Press, Cambridge, MA, 1987.
[2] P.R. Kotiuga, *Hodge decompositions and computational electromagnetics*, Ph.D. Thesis, Department of Electrical Engineering, McGill University, Montréal, 1984.
[3] Z. Ren, T-Ω formulation for eddy-current problems in multiply connected regions, IEEE Trans. Magn., Vol. 38, No. 2, pp. 557-560, 2002.
[4] P. Dłotko, R. Specogna, Cohomology in 3d magneto-quasistatics modeling, *Commun. Comput. Phys.*, Vol. 14, No. 1, pp. 48-76, 2013.
[5] A. Khebir, P. Dlotko, B. Kapidani, A. Kouki, R. Specogna, T-Ω formulation with higher order hierarchical basis functions for non simply connected conductors, Proceesings of the IEEE CEFC 2106, Miami.
[6] A. Khebir, P. Dlotko, B. Kapidani, A. Kouki, R. Specogna, T-Ω formulation with higher order hierarchical basis functions for non simply connected conductors, submitted, preprint available on arXiv:1704.03694 [physics.comp-ph].
[7] P. Dłotko, B. Kapidani, R. Specogna, TOPOPROCESSOR: an efficient computational topology toolbox for h-oriented eddy current formulations, *IEEE Trans. Magn.*, Vol. 53, No. 6, 7204404, 2017.
[8] M. Pellikka, S. Suuriniemi, L. Kettunen, C. Geuzaine, Homology and cohomology computation in finite element modeling, *SIAM J. Sci. Comput.*, Vol. 35, No. 5, pp. 1195-1214, 2013.
[9] P.R. Kotiuga, On making cuts for magnetic scalar potentials in multiply connected regions, J. Appl. Phys., Vol. 61, No. 8, pp. 3916-3918, 1987.
[10] P.R. Kotiuga, An algorithm to make cuts for magnetic scalar potentials in tetrahedral meshes based on the finite element method, *IEEE Trans. Magn.*, Vol. 25, No. 5, pp. 4129-4131, 1989.
[11] P.W. Gross, P.R. Kotiuga, *Electromagnetic Theory and Computation. A Topological Approach*, Cambridge University Press, New York, 2004.
[12] P. Dłotko, R. Specogna, Physics inspired algorithms for (co)homology computations of three-dimensional combinatorial manifolds with boundary, *Computer Phys. Commun.*, Vol. 184, No. 10, pp. 2257-2266, 2013.
[13] P. Dłotko, R. Specogna, Lazy cohomology generators: a breakthrough in (co)homology computations for CEM, *IEEE Trans. Magn.*, Vol. 50, No. 2, 7014204, 2014.
[14] R. Hiptmair, J. Ostrowski, Generators of $H_1(\Gamma_h, \mathbb{Z})$ for triangulated surfaces: construction and classification, *SIAM J. Comput.*, Vol. 31, No. 5, pp. 1405-1423, 2002.
[15] P. Dłotko, R. Specogna, Efficient generalized source field computation for h-oriented magnetostatic formulations, *Eur. Phys. J. Appl. Phys.*, Vol. 53, 20801, 2011.
[16] E. Tonti, *The Mathematical Structure of Classical and Relativistic Physics: A General Classification Diagram*, Birkhäuser, Basel, Switzerland, 2013.
[17] A. Alonso Rodriguez, E. Bertolazzi, R. Ghiloni, R. Specogna, Efficient construction of 2-chains with a prescribed boundary, *SIAM J. Numer. Anal.*, Vol. 55, No. 3, pp. 1159-1187, 2017.
[18] J.P. Webb, B. Forghani, A single scalar potential method for 3D magnetostatics using edge elements, *IEEE Trans. Magn.*, vol. 25, pp. 4126-4128, 1989.
[19] P.R. Kotiuga, Topological duality in threedimensional eddycurrent problems and its role in computeraided problem formulation, *J. Appl. Phys.*, Vol. 67, No. 9, pp. 4717-4719, 1990.
[20] B. Kapidani, P. Dłotko, P. Alotto, P. Bettini, R. Specogna, Computation of relative 1-cohomology generators from a 1-homology basis for eddy currents boundary integral formulations, *IEEE Trans. Magn.*, Vol. 52, No. 10, 7210006, 2016.
[21] R.L. Ricca, B. Nipoti, Gauss' linking number revisited, *J. Knot Theory Ramifications*, Vol. 20, pp. 1325-1343, 2011.
[22] J. Schoeberl, NETGEN An advancing front 2D/3D-mesh generator based on abstract rules, *Computing and Visualization in Science*, Vol. 1, No. 1, pp- 41-52, 1 997.
[23] P. Dłotko, R. Specogna, A novel technique for cohomology computations in engineering practice, *Comput. Methods Appl. Mech. Eng.*, Vol. 253, pp. 530-542, 2013.